DOCUMENT RESUME

ED 466 212 IR 021 283

AUTHOR	Schmidt, Peter
TITLE	Preparing Oral Examinations of Mathematical Domains with the Help of a Knowledge-Based Dialogue System.
PUB DATE	2001-00-00
NOTE	7p.; In: ED-Media 2001 World Conference on Educational Multimedia, Hypermedia & Telecommunications. Proceedings (13th, Tampere, Finland, June 25-30, 2001); see IR 021 194.
AVAILABLE FROM	Association for the Advancement of Computing in Education (AACE), P.O. Box 2966, Charlottesville, VA 22902 ($40, AACE members; $50, nonmembers). Tel: 804-973-3987; Fax: 804-978-7449; Web site: http://www.aace.org.
PUB TYPE	Reports - Descriptive (141) -- Speeches/Meeting Papers (150)
EDRS PRICE	MF01/PC01 Plus Postage.
DESCRIPTORS	*Calculus; *Computer Assisted Testing; Higher Education; *Instructional Design; *Mathematical Applications; *Mathematical Concepts; Mathematics Education; Teaching Methods
IDENTIFIERS	*Oral Examinations

ABSTRACT

A conception of discussing mathematical material in the domain of calculus is outlined. Applications include that university students work at their knowledge and prepare for their oral examinations by utilizing the dialog system. The conception is based upon three pillars. One central pillar is a knowledge base containing the collections of mathematical objects: theorems and their proofs, concepts, and task types. Manifold attributes complement the object specifications, including attributes related to categorization, didactics, applications, transition to other objects, and error handling. The second pillar consists of question types, dialogue strategies, and a user model. To formulate the answers, a restricted natural language is used. The selection and the order of the questions may be controlled by an explicit specification or implicitly by the choice of a dialogue strategy, including an exploration dialogue, a spot check dialogue, or a problem solving dialogue. The third pillar consists of procedural knowledge to monitor and verify the answers. For that, techniques of the fields of theorem providing and formula manipulation are employed. (Contains 11 references.) (Author/AEF)

Preparing Oral Examinations of Mathematical Domains with the Help of a Knowledge-Based Dialogue System

Peter Schmidt
Department of Computer Science, University of Bonn, Germany
email: peter@uran.informatik.uni-bonn.de

Abstract: A conception of discussing mathematical material in the domain of calculus is outlined. Applications include that university students work at their knowledge and prepare for their oral examinations by utilizing the dialogue system. The conception is based upon three pillars. One central pillar is a knowledge base containing collections of mathematical objects: theorems and their proofs, concepts, and task types. Manifold attributes complement the object specifications including attributes related to categorization, to didactics, to applications, to transition to other objects, and to error handling. The second pillar consists of question types, of dialogue strategies, and of a user model. To formulate the answers a restricted natural language is used. The selection and the order of the questions may be controlled by an explicit specification or implicitly by the choice of a dialogue strategy including an exploration dialogue, a spot check dialogue, or a problem solving dialogue. The third pillar consists of procedural knowledge to monitor and verify the answers. For that, techniques of the fields of theorem proving and of formula manipulation are employed.

Introduction

Mathematics is a part of many branches of study in the universities. Oral and written examinations are regular events which challenge the students, but also burden the students. The students employ a bunch of strategies to cope with the examinations. Success or failure of the students depend upon a large set of variables including the nature of the individuals, their learning history, the quality and conscientiousness of the instruction, and many more characteristics.

We here suggest conceptions of discussing mathematical material to serve several applications: University students may work at their knowledge and prepare for their oral examinations by utilizing the knowledge-based system. Or universities may use the system to replace oral or other written examinations. For that purpose, we describe the representation of the mathematical knowledge, a list of possible questions and the monitoring of the answers, some dialogue strategies to discuss the knowledge, and a scaffold of a student model. We chose the domain of calculus (see e.g. Smith 1983) to study dialogues because of the importance of calculus for the edifice of mathematics and for many practical applications and because calculus belongs to the first mathematical fields which are studied at the universities.

There are some typical characteristics of oral examinations which our conception takes into account: The subjects of an examination in the field of mathematics are the theorems, proofs, concepts, applications, and problem solutions of one or more domains. An examination should cover the whole material. Therefore the examiners employ typical dialogue strategies. Some of the topics are discussed in a detailed way and the rest of the material is covered by a series of spot check questions. The transition from one question to another is generally controlled by mathematical connections or by the answers of the student. The time which is spent on the detailed exploration of a topic essentially depends on the performance of the examinee. When the student shows knowledge gaps or insecurities with a topic the examiner will further inquire to clarify the background knowledge and abilities of the student, e.g. by sticking to the the central points, by asking additional questions related to the foreknowledge, or by giving another problem to solve. In that case, the reaction of the examiner is strongly triggered by the answers. Usually, the rating of the examinee's performance does not only depend on the ratio of correctly answered questions and wrongly answered questions, but also on the flexibility and easiness with which the student deals with the questions and hints during the dialogue.

Mathematical Knowledge Base

The mathematical knowledge groups around the following objects: concepts, theorems, and task types. The specification of the objects includes a series of attributes referring to the object itself and its use of holding a dialogue. We will describe the representation of theorems in a more detailed way and mention only a few details related to concepts and task types because of space limitations.

To word the premises, the conclusions, and the proof(s) of a theorem we employ a restricted natural language (see Schmidt 2000) or the website (see Schmidt 2001).

Representation of Theorems

The specification of a theorem consists of a series of attributes related to various information: Object describing information, classifying information, didactical information, application related information, transitional information, and error information. In the following we describe the attributes in the context of the specification of the mean-value theorem (Fig. 1). To represent the mathematical knowledge we use XML (see e.g. Goldfarb & Prescod 1998). The informal representation of figure 1 is chosen because of readability.

```
THEOREM
   DOMAIN: calculus
   SUBDOMAINS: differentiable functions
   CATEGORY: existence of a point
   PRIORITY: high
   FOREKNOWLEDGE: continuous, differentiable
   NAME: mean-value theorem
   PREMISES:  f: [a,b] -> R
              f IS continuous IN [a,b]
              f IS differentiable IN (a,b)
                 ERROR: (CONDITION "f IS differentiable IN [a,b]",
                         REACTION  "IN [a,b] is not necessary.")
   CONCLUSIONS: SOME x0 IN (a,b):  f(b)-f(a) = (b-a)*f'(x0)
   USES: (THEOREM, CONCLUSIONS, "f IS monotonously increasing IN[a,b]")
   TRANSITION: (THEOREM, generalized mean-value theorem,
                 FORMULATE(THEOREM, generalized mean-value theorem),
                 "There is a generalization of the mean-value theorem.")
   TRANSITION: (THEOREM, mean-value theorem IN R^n, ...)
```

Figure 1: Partial specification of a theorem

Object describing information. The object describing information consists of a series of attributes including the external name (if one exists) of the object, its internal name, the premises, the conclusions, and the proof(s). See the example of the figure 1.

Classifying information. Each object belongs to a domain and possibly to one or more subdomains and subsubdomains. The theorem above e.g. belongs to the subdomain of the differentiable functions. The theorems are moreover classified with regard to their conclusions; the category of the mean-value theorem is e.g. *existence of a point*. The underlying idea of that classification is that the proofs of the theorems belonging to the same category often employ a similar proof strategy and utilize related heuristics. That information becomes relevant in the context of proof related questions. Further categories of theorems among about a dozen others are e.g. *property of a function* and *uniqueness of a value, a point or a function*.

Didactical information. Didactical information among others refers to the foreknowledge and to the priority of the object. The information regarding the priority which is high in the case of the mean-value theorem is utilized to control the selection of questions.

Application related information. Application related information employs two attributes. The attribute *uses* states the use of the theorem with respect to proofs of other theorem. The example of the figure 1 shows that the mean-value theorem is used to prove a theorem which concludes about a function being monotonously increasing. The other attribute *applications* refers to practical applications.

Transitional information. Transitional information refers to the order of questions and successors of questions in a dialogue. Transitions from one object or attribute to another are explicitly or implicitly defined by the knowledge base. Implicit transitions are e.g. triggered by the premises or the conclusions of a theorem. The specification of the figure 1 contains two explicit transitions to two other theorems, a generalization of the mean-value theorem and a version of the theorem in the case of n dimensions. The attribute *transition* takes four parameters: The object classification, the name of the object, the question (see below), and possibly a text which is used to word the transition in a dialogue.

Error information. Error information may be attached to some attributes of a specification. That information serves the purpose of specifically treating expected errors. Simple error information takes two parameters: The first one specifies the error and the second one words the reaction of the system. The *error* attribute of the figure 1 refers to the preceding statement "f IS differentiable IN (a,b)". See the discussion of errors below. Error information may be nested to cope with succeeding errors.

Representation of Concepts and Task Types

The specification of a *concept* consists of a series of attributes including among others its name , the premises, the defining statements, examples, and counter-examples.

A mathematical *task type* covers tasks which may be solved by the same or by similar methods. With the specification, we make a distinction between formula manipulation task types (e.g. the calculation of a limit or the solution of a differential equation and word problem types (e.g. extremal value problems which describe a geometrical situation). The tasks of both kinds of task types are on the top level classified into *subtypes* which may be characterized by certain patterns, by the solution methods, or by the values of certain relevant parameters of the task type.

The Questions and the Monitoring of the Answers

The various types of questions employ a certain format. There are several attributes attached to the question types including the question template, the answer directions, and possibly an answer template. In general, the questions of the system are presented in a natural language and the students word their answers by using the above-mentioned restricted natural language.

Questions Related to Theorems and Proofs

There is a list of standard question types which refer to the premises, conclusions, proofs, uses, and applications of a theorem. The following list of question types shows their formal statements. The formal statement of a question type is among others used in the context of specifying a dialogue (see below). The program system employs an XML representation as with the knowledge base.

(a) FORMULATE (THEOREM, theorem name)
(b) FORMULATE (PREMISES, statement)
(c) FORMULATE (CONCLUSIONS, statement)
(d) FORMULATE (USES, theorem name)
(e) FORMULATE (APPLICATIONS, theorem name)
(f) FORMULATE (PROOF, theorem name)

The question type **(a)** refers to the formulation of a theorem when its name is given. An example of a concrete question is "Formulate the mean-value theorem". The *answer template* consists of the words "mean-value theorem", "PREMISES", and "CONCLUSIONS". The user has to particularize the premises and the conclusions by using the above-mentioned restricted natural language. The *answer directions* consist of the format of the statements which may be used to word a theorem.

The question type **(b)** refers to the formulation of premises from which a given conclusion may be derived. An example of a concrete question is "Which premises are sufficient so that the function f:[a,b] -> R is monotonously increasing in [a,b]?".

The question type **(c)** refers to the formulation of conclusions which may be inferred, if given premises are valid. An example of a concrete question is "The function f:[a,b] -> R is continuous in [a,b]. Which properties of the function f may be inferred?

The question types **(d)** and **(e)** refer to the listing of uses and of applications of a theorem. An example of a concrete question is "Give applications of the theorem of Taylor."

The question type **(f)** refers to the formulation of a proof. An example of a concrete question is "Prove: If f is differentiable at a , then f is continuous at a."

The above-mentioned standard question types may be modified so that the answers are shorter, more specific, or easier. Some examples are: With **(a)** some parts of the theorem may be provided so that the wording of the theorem must only be completed. With **(f)** the question may refer to the proof ideas or to completing a given proof by adding e.g. lacking foundations or lacking statements. We here omit further details.

Monitoring and Verifying the Answers Related to Theorems and Proofs

To monitor and to verify the answers of a user the system employs on the one hand the knowledge base of concepts, theorems, proofs, and task types and on the other hand methods of theorem proving and of formula manipulation. In the following we assume that the reader is familiar with the basic concepts of theorem proving, especially with the transformation of a logical formula into a quantifier free form and with the process of unification (see e.g. Bibel 1987 or Chang & Lee 1973). Regarding the theorem proving techniques we utilize methods which are similar to the methods of Bledsoe, Boyer, and Henneman to automatically prove limit theorems (Bledsoe et al. 1972).

An important goal is that correct answers are recognized as correct by the system. It is known that theoretical limitations restrict the possibility of deciding whether an answer is correct (see e.g. Richardson 1968 and Bibel 1987), but such cases here are of a minor importance. The aim here is to also accept alternative, correct answers which are not identical to the correct answers which are calculated, derived, or stored by the system. One cannot expect that a computer program could compete with a human in the field of intelligently treating the answers of a user. Nevertheless a detailed knowledge base and the methods of theorem proving and of formula manipulation allow for a useful and reliable treatment of the answers.

Earlier research in the domain of Intelligent Tutoring Systems suggests that competent error handling is a difficult and tedious task even in the case of apparently simple procedural problem solving (see e.g. Wenger 1987). The framework provided by Collins and Stevens who list 24 rules in the context of the Socratic method of tutoring is helpful with respect to classifying general errors like e.g. overgeneralization bugs or overdifferentiation bugs (Collins 1977; Stevens & Collins 1977). When an answer is recognized as not being correct, several mechanisms may be applied to diagnose the error and to react in an appropriate way. The mechanisms which are used depend on the question types, on the contents of the knowledge base including the specification of errors within the concrete objects, and on the methods which are provided by the logic subsystem and by the formula manipulation subsystem. The specificity of the possible feedback of course depends on the detailedness of the knowledge base and on the error classifying possibilities of the subsystems.

In the following we will give some examples of how the answers are diagnosed and treated.

Monitoring the questions of the question type (a). To check a user's formulation of a theorem the system employs the theorem of the knowledge base. The answer is correct when the user's premises and conclusions correspond to those of the theorem in the knowledge base. The statements are compared using a quantifier free form and the process of unification.

Some general errors which may occur are: A premise or a conclusion is lacking, a premise is not sufficient or not necessary, and the conclusion is too weak or too strong. Assume that the question refers to the mean-value theorem and the user enters "f IS differentiable IN [a,b]" as a premise instead choosing the open interval (a,b). That error can be treated by utilizing the specification of the error in the object mean-value theorem (Fig. 1). The reaction of the system succeeds the error condition (Fig. 1). In the case that the error is not specified with the mean-value theorem, the error would have been detected by comparing the user's answer with the correct answer and by establishing that the statement "f IS differentiable IN [a,b]" subsumes the statement "f IS differentiable IN (a,b)" and so is not necessary.

Monitoring the questions of the question type (b). The basis of checking the premises from which a given conclusion may be derived are the theorems of the knowledge base which infer that conclusion. When a user e.g. enters the premises P1, P2, and P3 the system checks whether there is a theorem in the knowledge base with the premises P1, P2, and P3 and the stated conclusion. Similar considerations as in the case of **(a)** apply.

Monitoring the questions of the question type (c). Assume that the premise P is given and the knowledge base contains several theorems inferring the conclusions C1, C2, ..., Cn from P. When the user enters one of the C's the answer is correct. Assume the user enters D which does not correspond to one of the C's. Then the theorem "P implies D" might be wrong or correct. In the correct case, the theorem is likely a marginal one, because it is not a part of the knowledge base. In the wrong case, certain errors can be detected by checking whether a theorem "D implies P" is a part of the knowledge base. If "D implies P" is a valid theorem and D and P are not equivalent, there will probably be a counter-example in the knowledge base. An concrete example of that case is the wrong conclusion "f is continuous at x0 implies f is differentiable at x0".

Monitoring the questions of the question type (d) and *(e)*. The questions referring to the uses and applications of a theorem mainly involve keywords as answers. Providing a list of synonyms is sufficient in most cases. More difficult questions may be handled by multiple choice questions.

Monitoring the questions of the question type (f). The monitoring of a user's proof is discussed in (Schmidt 2000).

Absurd answers. A human examiner cannot only check the correctness of an answer, but also the degree with which an answer is sensible in contrast to being foolish or inadequate. The worth and the assessment of a system increases when the system can eliminate absurd answers by a corresponding reaction. One such method is based on decomposing an answer into its syntactical parts and checking whether they correspond to the correct answer. In the case of the mean-value theorem the conclusion consists of an existential quantification which is followed by a formula containing the quantified variable. An answer which does not fulfil that restriction is regarded as being not appropriate and is treated by a corresponding comment. Since the XML representation of the objects covers the decomposing, such syntactical means may be applied without adding further knowledge elements to the mathematical objects.

Questions Related to Concepts and Task Types

We here just mention some question types related to concepts and task types. We omit further information because of the lack of paper space.

The question types related to concepts are: (i) The formulation of the definition of a concept, (ii) stating examples of the concept, (iii) stating counter-examples of the concept, (iv) deciding whether a given object is an example of the concept, (v) stating equivalent definitions, and (vi) stating sufficient conditions which imply the concept.

The questions related to task types are: (i) Calculation or solution of a formula manipulation task (e.g. calculation of a limit, an integral, the solution of a differential equation). (ii) classification of a task and stating the solution method, (iii) listing classes of tasks holding a certain property, and (iv) setting up initial equations to solve a problem which is stated in a natural language (e.g. an extremal value problem).

Dialogues

The Selection and the Order of the Questions in a Dialogue Session

A dialogue consists of a series of questions and answers. The selection and the order of the questions on the top level may be controlled by three different mechanisms: (i) By an external explicit specification of the subjects of the dialogue, (ii) by several dialogue strategies which are offered by the system, and (iii) by an explicit choice of single questions by the user. In the following we briefly describe the three mechanisms.

(i) Explicit specification of the subjects of the dialogue. The selection of the questions and their order may be explicitly specified in a DIALOGUE statement. Specifications of typical dialogues are provided by the system. They may be easily joined to form a complete session. This mechanism is useful when there is information about which knowledge is relevant in the current situation. The example of figure 2 shows a part of a dialogue specification (the program system employs a corresponding XML representation). There is a sequence of questions at first asking the definition of differentiable, secondly inquiring after a counter-example of a not differentiable function, and thirdly asking the formulation of the theorem of Taylor.

```
DIALOGUE
  FORMULATE (DEFINITION, "f IS differentiable AT x0")
  FORMULATE (COUNTER-EXAMPLE, "f IS differentiable AT x0" )
  FORMULATE (THEOREM, theorem of Taylor)
```

Figure 2: Explicit specification of a dialogue

(ii) Dialogue strategies which are offered by the system. The objects which the knowledge base contains automatically trigger questions (e.g. the above-mentioned questions of the question types) and transitions to questions related to other objects (e.g. by the specified transitions or by the specified or implicitly given foreknowledge). For an implicit selection of questions we currently distinguish between the following three dialogue strategies which take into account the characteristics of oral examinations:

(a) A *spot check dialogue* relates to a domain, subdomain or subsubdomain and chooses questions so that all the parts of the subdomain are essentially covered by the questions. A corresponding statement is e.g. SPOT CHECK DIALOGUE (SUBDOMAIN, continuous functions) which triggers questions related to the subdomain of continuous functions. By the choice of additional parameters one may definitely include certain topics (e.g. all the theorems with a high priority) or exclude certain topics (e.g. task types or transitions to objects to other subdomains).

(b) An *exploration dialogue* tries to explore one or more objects or a small subdomain in a systematic way to a certain degree. E.g. the statement EXPLORATION DIALOGUE (THEOREM, mean-value theorem) triggers the questions which are explicitly or implicitly related to the mean-value theorem. By the selection of parameters one can include or exclude certain questions.

(c) A *problem solving dialogue* refers to a task type. It consists of solving problems and of answering questions related to the methods which are used to solve such problems. An example is PROBLEM SOLVING DIALOGUE (TASK TYPE , ordinary differential equations).

(iii) Explicit choice of the user. The user can traverse the objects of the knowledge base and get a list of the questions and problems from which he or she can choose anyone.

The Student Model

The scaffold of a student model is based upon the subdomains and their objects. The model stores information regarding three general aspects. *One aspect* is the material which is relevant to the student. The student may explicitly or implicitly (by selecting related questions) choose the subdomains or the objects. The *second aspect* is the self-assessment of the own knowledge. A student may state his or her own understanding or performance (good, medium, poor) with the subdomains, with single objects, and with the general categories of theorems, proofs, concepts, and solving tasks. Whereas the utility and the use of diagnostic processes are main subjects with tutoring systems, the topic of self-assessment is widely neglected as a basis of tutorial actions. That is surprising since the students may in most cases realistically assess their mathematical abilities and they usually know when they do not understand a subject and they know their deficiencies. The *third aspect* refers to the concrete dialogues and the student's performance. A trace of the details of the dialogue (e.g. questions, hints of the system, errors) is stored and abstracted to form a view of a student's knowledge and ignorance of the material of the subdomains and of the theorems, the concepts, and the task types. Another factor is the user's ability of correcting errors or utilizing hints. The contents of the user model may influence the progress of the dialogue: Wrong answers may change the order of questions (e.g. it is not sensible to discuss a concrete application of a theorem when the user does not know the theorem) or trigger additional questions (e.g. an answer may show that a user is not familiar with some foreknowledge; the system would then put some questions related to the foreknowledge).

Summary and Outlook

We outlined a conception of a knowledge-based dialogue system dealing with mathematical material in the domain of calculus. Such a dialogue system may be employed in the context of a virtual university or of a face-to-face university and the applications may include the preparation of oral examinations.

Our current prototype includes an interface to print the questions and enter the answers, some procedures of theorem proving and an own formula manipulation system. The programming language is Java including the servlet API. The program system is developed as a web application. The knowledge is represented using XML. The prototype will be further developed with respect to the methods and to the knowledge base.

References

Bibel, W. (1987). *Automated Theorem Proving*.Vieweg, Braunschweig, Germany.

Bledsoe, W.W., Boyer, R.S. , Henneman, W.H. (1972). Computer proofs of limit theorems. *Artificial Intelligence* 3, 27-60.

Chang,C.-L. & Lee, R.C. (1973). *Symbolic Logic and Mechanical Theorem Proving*. Academic Press, New York, 1973.

Collins, A. (1977). Processes in Acquiring Knowledge, in: Anderson, Spiro, Montague (Editors), *Schooling and the Acquisition of Knowledge*, Lawrence Erlbaum Ass., Hillsdale, N.J.

Goldfarb, C. F. & Prescod, P. (1998). *The XML Handbook*. Prentice Hall PTR, Upper Saddle River, NJ, USA.

Richardson, D. (1968). Some Undecidable Problems Involving Elementary Functions of a Real Variable. *The Journal of Symbolic Logic*. 33, 514-520.

Schmidt, P. (2000). Monitoring and Verifying Mathematical Proofs Formulated in a Restricted Natural Language. *Computers in Education*, 2000, (ICCE 2000), National Tsing Hua University, Taiwan. 1315-1323.

Schmidt, P. (2001). The restricted natural language to word theorems and proofs. http://www.cs.uni-bonn.de/~peter/EDMEDIA2001supplement.html

Smith, K.T. (1983). *Primer of Modern Analysis*. Springer Verlag, New York, USA.

Stevens, A.L. & Collins, A. (1977). The Goal Structure of a Socratic Tutor. *Annual Conf. Assoc. for Comp. Machinery*, 1977, 256-263.

Wenger, E. (1987). *Artificial Intelligence and Tutoring Systems, Computational and Cognitive Approaches to the Communication of Knowledge*. Morgan Kaufmann Publishers, Inc.,Los Altos, California, USA.

NOTICE

Reproduction Basis

☒ This document is covered by a signed "Reproduction Release (Blanket)" form (on file within the ERIC system), encompassing all or classes of documents from its source organization and, therefore, does not require a "Specific Document" Release form.

☐ This document is Federally-funded, or carries its own permission to reproduce, or is otherwise in the public domain and, therefore, may be reproduced by ERIC without a signed Reproduction Release form (either "Specific Document" or "Blanket").